Freeman 曲線を用いた Xate および R–ate ペアリングのための定義体における乗算アルゴリズム

根角 健太 † 湯浅 達也 ‡ 野上 保之 † 森川 良孝 †

† 岡山大学大学院自然科学研究科
‡ 岡山大学工学部通信ネットワーク工学科

700-8530 岡山県岡山市北区津島中 3-1-1
{nekado,yuasa,nogami,morikawa}@trans.cne.okayama-u.ac.jp

あらまし 近年，ID ベース暗号やグループ電子署名に対して，ペアリングをベースに実装する方式が注目を集めている．これらの方式を高速化するためには，ペアリングアルゴリズムの改良だけではなく，定義体における演算も高速化することが重要である．そこで，著者らは効果的にペアリングベース暗号を実装することができる Freeman 曲線に注目し，SCIS2009 にて Freeman 曲線のための拡大体の構成法について考え，その拡大体上演算の高速化手法について提案した．本稿では，SCIS2009 で提案したものよりも高速な手法を追求し，その手法を用いたときの Xate および R–ate ペアリングの実装結果を示す．

Multiplication Algorithms in The Extension Field for Xate and R–ate Pairings with Freeman Curve

Kenta Nekado† Tatsuya Yuasa‡ Yasuyuki Nogami† Yoshitaka Morikawa†

†Graduate School of Natural Science and Technology, Okayama University
‡Communication Network Engineering, Okayama University

3-1-1 Tsushima-naka, Okayama, Okayama, 700-8530, Japan
{nekado,yuasa,nogami,morikawa}@trans.cne.okayama-u.ac.jp

Abstract Recently, pairing–based cryptographies such as ID–based cryptographic and group signature have been studied. For their implementations, pairings such as Xate pairing and R–ate pairing have been efficient. In order to make pairing calculations faster, it is important to make not only pairing algorithms but also arithmetic operations in the defined field efficient. Thus, in several kinds of ordinary pairing–friendly curves, we focused on Freeman curve that enables implementations of fast pairing–based cryptographies, and proposed efficient implementation methods of arithmetic operations in the definition field, in SCIS2009. This paper proposes more efficient methods, and shows implementation results of Xate pairing and R–ate pairing.

1 はじめに

近年，ID-based 暗号 [1] やグループ電子署名 [2] に対して，ペアリングをベースに実装する方式が注目を集めている．このペアリングには様々な種類が存在し，高速に計算可能なペアリングとして，例えば Ate ペアリングを改良した Xate ペアリング [3] や R–ate ペアリング [4] といったものが提案されている．ペアリングを実装

するためには，ペアリング親和曲線が必要不可欠であり，現在までに数多くの曲線が提案されている．その中でも，$\rho = \lfloor \log_2 p \rfloor / \lfloor \log_2 r \rfloor \approx 1$（$p$：定義体の標数，$r$：楕円曲線の位数）を満たす曲線がペアリング実装に適しているとされており，その条件を満たす曲線として Miyaji-Nakabayashi-Takano (MNT) 曲線 [5]，Barreto-Naehrig (BN) 曲線 [6]，Freeman 曲線 [7, 8] といった非超特異ペアリング親和曲線が提案されている．これらの曲線の定義体には，世界中の研究者の多くが四則演算が高速な optimal extension field (OEF) [9] を用いる．しかし，Freeman 曲線がもつパラメータの制限により，OEF の手法だけでは Freeman 曲線の定義体を準備することはできない．一方で，加藤らが提案している type–X all–one polynomial field (AOPF) [10] は OEF とほぼ同様に四則演算を高速に行うことができ，さらに Freeman 曲線の定義体に用いることができる．

そこで，著者らは SCIS2009 [11] にて，Freeman 曲線の定義体を type–X AOPF で準備し，主に type–X AOPF の乗算アルゴリズムを準備した定義体用に最適化することで，乗算，自乗算および逆元計算の高速化を行った．そのときの最適化では type–X AOPF がもつ巡回性を巧みに利用することで成し得たのだが，本稿ではさらに木構造を用いた処理を行うことでさらなる最適化を目指す．加えて，本稿では Freeman 曲線を用いた Xate ペアリングと R–ate ペアリングの実装を行い，Xate ペアリングが R–ate ペアリングとほぼ同等に効率の良いペアリングであることを示す．

以下とくに断らない限り，p および m を素数（標数）および正整数とし，$\mathbb{F}_p, \mathbb{F}_{p^m}$, および $\mathbb{F}_{p^m}^*$ を素体，m 次拡大体および m 次拡大体の乗法群とする．$a \,|\, b$ あるいは $a \nmid b$ は a が b を割り切るあるいは割り切らないことを表す．また，A_n, D_n, M_n, S_n および I_n は $\mathbb{F}_{p^n}$ 上の加算，二倍算，乗算，自乗算および逆元計算のコストとする．

2 数学的準備

本章では，各種ペアリングと Freeman 曲線について復習する．ただし，$E(\mathbb{F}_{p^m})$ と $E'(\mathbb{F}_{p^m})$ を $\mathbb{F}_{p^m}$ 上で定義される楕円曲線上の有理点が成す加法群とそのツイスト曲線上の有理点が成す加法群とし，t を $E(\mathbb{F}_{p^m})$ のフロベニウスのトレースとする．また，$[i]$ は有理点の i 倍の写像を表し，$E(\mathbb{F}_{p^m})[r]$ は $E(\mathbb{F}_{p^m})$ に含まれる素数位数 r の有理点の集合を表す．

2.1 Ate ペアリング

$r|(p^k - 1)$ を満たす最小の正整数 k をペアリングにおける埋め込み次数とし，$\mathbb{G}_1$ と $\mathbb{G}_2$ を

$$\mathbb{G}_1 = E(\mathbb{F}_{p^k})[r] \cap \mathrm{Ker}(\phi - [1]), \tag{1a}$$

$$\mathbb{G}_2 = E(\mathbb{F}_{p^k})[r] \cap \mathrm{Ker}(\phi - [p]), \tag{1b}$$

とすると，Ate ペアリング ϵ は非退化な双線形写像として，次式のように定義される．

$$\epsilon : \begin{cases} \mathbb{G}_2 \times \mathbb{G}_1 & \to \ \mathbb{G}_3 = \ \mathbb{F}_{p^k}^* / (\mathbb{F}_{p^k}^*)^r \\ (Q, P) & \mapsto \ f_{T,Q}(P)^{(p^k-1)/r} \end{cases}, \tag{2a}$$

$$T \equiv (t-1)^i \mod r \quad (1 \le i < k). \tag{2b}$$

Ate ペアリングは Miller のアルゴリズムにより $f_{T,Q}(P)$ を計算するステップと最終べきとよばれる $f_{T,Q}(P)$ の $(p^k - 1)/r$ 乗を計算するステップの 2 つのステップで構成される．Ate ペアリングでは Miller のアルゴリズムにおける計算処理の繰り返し回数は $\lfloor \log_2(T) \rfloor$ となる．

Devegili らは BN 曲線を用いたペアリングを効果的に実装するために，部分体ツイスト曲線を用いて Ate ペアリングの高速化を行っている [12]．一方，野上らも同様に部分体ツイスト曲線を活用することで Ate ペアリングの高速化を図っている．この改良された Ate ペアリングは cross–twisted Ate (Xt–Ate) ペアリングと呼ばれる [3]．Devegili らの手法ではペアリングを効果的に実装するために，その定義体を多項式基底で構成する必要がある．一方，野上らの手法ではそのような制約は存在せず，6 次ツイスト曲線を用いた場合には Devegili らの手法にわずかに劣るが，2 次，3 次，4 次ツイスト曲線を用いる場合にはわずかに優れている．Freeman 曲線では 2 次ツイスト曲線しか用いることができないため，本稿では部分体ツイスト曲線を用いた手法として野上らの手法を採用する．

2.2 Freeman 曲線

Freeman 曲線 [7, 8] は埋め込み次数が 10 の非超特異ペアリング親和曲線であり，そのパラ

メータは次式で与えられる．ただし，χ は $p(\chi)$ が素数となるような正整数である．

$$p(\chi) = 25\chi^4 + 25\chi^3 + 25\chi^2 + 10\chi + 3, \quad (3a)$$
$$r(\chi) = 25\chi^4 + 25\chi^3 + 15\chi^2 + 5\chi + 1, \quad (3b)$$
$$T(\chi) = (t(\chi) - 1)^2 = 5\chi^2. \quad (3c)$$

Freeman 曲線 $E(\mathbb{F}_{p^{10}})$ を用いた Xt–Ate ペアリグではその 2 次ツイスト曲線 $E'(\mathbb{F}_{p^5})$ を活用する．ゆえに，このペアリングを高速化するためには定義体 $\mathbb{F}_{p^{10}}$ 上の演算を高速化するだけではなく，その部分体 $\mathbb{F}_{p^5}$ 上の演算も高速化する必要がある．

2.3 改良された Ate ペアリング

野上らは整数変数 χ を用いて改良した Ate ペアリング（*integer variable* χ*–based* Ate, Xate ペアリング）を提案している．Xate ペアリングもまた非退化な双線形写像であり，Freeman 曲線を用いた Xate ペアリング $\hat{\epsilon}$ は次式のように定義される．ただし，l_{Q_1,Q_2} は Q_1，Q_2 を通る直線を示す．

$$\hat{\epsilon} : \begin{cases} \mathbb{G}_2 \times \mathbb{G}_1 \ \to \ \mathbb{G}_3 = \ \mathbb{F}_{p^k}^* / (\mathbb{F}_{p^k}^*)^r \\ (Q, P) \quad \mapsto \ \hat{f}_{\chi,Q}(P)^{(p^k-1)/r} \end{cases}, \quad (4a)$$

$$\hat{f}_{\chi,Q}(P) = \left(f_{\chi,Q}(P)^{(1+p)} \cdot l_{\chi Q,\, p\chi Q}\right)^{1+p^3} \cdot l_{(1+p)\chi Q,\, p^3(1+p)\chi Q}. \quad (4b)$$

式(4b)に示すように，Xate ペアリングの Miller のアルゴリズムにおける計算処理の繰り返し回数はほぼ $\lfloor \log_2(\chi) \rfloor$ となり，Ate ペアリングのときに比べるとほぼ 1/2 回に減少する．

一方で，Lee らもまた Miller のアルゴリズムにおける計算処理の繰り返し回数をほぼ $\lfloor \log_2(\chi) \rfloor$ に落とした Ate ペアリング（R–ate ペアリング）を提案している [4]．R–ate ペアリングもまた非退化な双線形写像であり，Freeman 曲線を用いた R–ate ペアリング $\tilde{\epsilon}$ は次式のように定義される．ただし，$a = -5\chi - 3$ である．

$$\tilde{\epsilon} : \begin{cases} \mathbb{G}_2 \times \mathbb{G}_1 \ \to \ \mathbb{G}_3 = \ \mathbb{F}_{p^k}^* / (\mathbb{F}_{p^k}^*)^r \\ (Q, P) \quad \mapsto \ \tilde{f}_{a,Q}(P)^{(p^k-1)/r} \end{cases}, \quad (5a)$$

$$\tilde{f}_{a,Q}(P) = f_{a,Q}(P)^{1+p^3} \cdot f_{2,Q}(P) \cdot l_{aQ,\,2Q} \cdot \ l_{p^2aQ,\,(a+2)Q}. \quad (5b)$$

これらの改良された Ate ペアリングにはそれぞれ Xt–Ate ペアリングの手法を適用することができ，本稿ではそれぞれを Xt–Xate ペアリングおよび Xt–R–ate ペアリングと呼ぶ．

3 Freeman 曲線の定義体

Freeman は [7, 8] で p および r の bit 長が異なる 4 種類の曲線を示している．本稿では，これらの曲線を中心に Freeman 曲線の定義体について考える．ただし，本稿では拡大体における逆元計算アルゴリズムとして，拡大体における乗算と Frobenius 写像によって構成される伊東–辻井アルゴリズム（ITA）[13] を採用する．

3.1 多項式基底で構成される拡大体

Bailey らは拡大次数 m の各素因数が $p-1$ を割り切るときに構成できる拡大体 optimal extension field (OEF) [9] を提案している．OEF は多項式基底により構成され，既約二項式を法多項式とする．OEF 上では四則演算を高速に行うことができるため，世界中の研究者の多くが定義体として用いる．しかし，式(3a)を考慮すると $5 \nmid (p(\chi)-1)$ であるため，OEF を Freeman 曲線の定義体 $\mathbb{F}_{p^{10}}$ およびその部分体 $\mathbb{F}_{p^5}$ として用いることはできない．一方，式(3a)を満たす OEF $\mathbb{F}_{p^2}$ は法多項式を $x^2 - 1$ として必ず構成することができる．したがって，部分体 $\mathbb{F}_{p^5}$ を OEF とは別の拡大体で準備し，それを OEF の構成手法を用いて 2 次逐次拡大することにより，Freeman 曲線の定義体 $\mathbb{F}_{(p^5)^2}$ を構成することができる．そこで，本稿では 2 次逐次拡大に OEF の手法を採用する．OEF $\mathbb{F}_{(p^5)^2}$ における演算のコストを表 1 に示す．

表 1: OEF $\mathbb{F}_{(p^5)^2}$ における演算のコスト

乗算・自乗算アルゴリズム	Schoolbook 法 [14]	Karatsuba 乗算 [14] Complex 自乗算 [14]
乗算	$(4, 0, 20, 0)^\dagger$	$(3, 0, 4, 0, 0)^\dagger$
自乗算	$(2, 1, 1, 1, 0)^\dagger$	$(2, 1, 3, 0, 0)^\dagger$
逆元演算	$(4, 0, 1, 0, 1)^\dagger$	

‡ (a, b, c, d, e) は $aM_5 + bS_5 + cA_5 + dD_5 + eI_5$ を示す．

多項式基底で構成される拡大体の中で OEF の次に四則演算が高速な拡大体は法多項式に既約三項式を用いるものである．本稿では，この拡大体を trinomial–based field (TBF) と呼ぶ．

Freeman 曲線における定義体の部分体 $\mathbb{F}_{p^5}$ として利用できる TBF の法多項式の例を表 2 に示す．TBF は OEF に比べて，乗算や自乗算のコストは大差がないが，Frobenius 写像のコストが大きい．TBF $\mathbb{F}_{p^5}$ における演算のコストを表 3 に示す．

表 2: TBF $\mathbb{F}_{p^5}$ の法多項式

p=149–bit	p=196–bit	p=234–bit	p=252–bit
x^5-4x-4	x^5-x-2	x^5-x-1	x^5-x-4

3.2 正規基底で構成される拡大体

著者らは SCIS2009 で Freeman 曲線の定義体およびその部分体 $\mathbb{F}_{p^5}$ として type–X all–one polynomial field (AOPF) [10] を用いることを提案した．そこで本節では，Freeman 曲線における定義体の部分体をどのようにして type–X AOPF で準備するのかを復習し，その構成した部分体における乗算の高速化を SCIS2009 のときと少し異なるアプローチで考える．

3.2.1 Type–$\langle h, m\rangle$ GNB

Type–X AOPF の構成には Gauss period normal basis (GNB) [15] を用いる．GNB は正整数 h を用いて次のように定義される．

定義 1： $\mathbb{F}_{hm+1}$ の 1 の原始 h 乗根を θ として，$\gamma = \sum_{i=0}^{h-1} \beta^{\theta^i}$ による共役元の集合 $\{\gamma, \gamma^p, \cdots, \gamma^{p^{m-1}}\}$ は $\mathbb{F}_{p^m}$ において正規基底を成す．ただし，β は $\mathbb{F}_{p^e}$ における 1 の原始 $km+1$ 乗根である．本稿では，この正規基底を type–$\langle h, m\rangle$ GNB と呼ぶ． □

Type–$\langle h, m\rangle$ GNB は $4p \nmid m(p-1)$ のとき必ず構成できる [10] ため，type–X AOPF は $p > m$ を満たす任意の (p, m) の組に対して必ず構成できる．したがって，Freeman 曲線における定義体の部分体 $\mathbb{F}_{p^5}$ も必ず構成することができ，そのときの最小の h は表 4 のようになる．

表 4: Type–X AOPF $\mathbb{F}_{p^5}$ を構成できる最小の h

p	149–bit	196–bit	234–bit	252–bit
h	2	6	8	6

3.2.2 Type–$\langle h, m\rangle$ CVMA

Type–X AOPF における乗算に適したアルゴリズムとして，加藤らは *cyclic vector multiplication algorithm* (CVMA) [10] を提案している．本稿では，type–$\langle h, m\rangle$ GNB で構成された type–X AOPF における CVMA を特に type–$\langle h, m\rangle$ CVMA と呼ぶ．Type–$\langle h, m\rangle$ CVMA による乗算および自乗算のコストは次式で与えられる．ただし，H_n を $\mathbb{F}_{p^n}$ 上の h 倍算とする．

$$M_m = \frac{m(m+1)}{2} M_1 + \begin{cases} \left(\left(\frac{m(m-1)(h+2)}{2} - 1\right) + m\right) A_1 + H_1 & (h : \text{奇数}) \\ \left(\frac{m(m-1)(h+2)}{2}\right) A_n & (h : \text{偶数}) \end{cases} \quad (6a)$$

$$S_m = M_m - (m(m-1)/2) A_1 \quad (6b)$$

式 (6a) に示すように，$\mathbb{F}_{p^m}$ における CVMA は h が大きくなるにつれ，$\mathbb{F}_p$ 上の加算をより多く必要とする．$\mathbb{F}_p$ 上の加算のコストは $\mathbb{F}_p$ における乗算のコストに比べて小さいため，$\mathbb{F}_{p^m}$ における CVMA で必要となる $\mathbb{F}_p$ 上の加算コストの総計はある程度許容できるが，あまりにも加算回数が多い場合，加算コストの総計は許容できないものとなる．

定義 1 より，type–X AOPF $\mathbb{F}_{p^5}$ を構成することができる最小の h は 2 といえるが，Freeman 曲線を用いる場合，最小の h が 2 となるときは表 4 に示すように p が 149–bit のときのみである．したがって，それ以外の場合は $\mathbb{F}_p$ における加算回数が $h = 2$ に比べて多くなり，乗算のコストが高くなってしまっていた．そこで，著者らは SCIS2009 で p が 196–bit の場合の CVMA による乗算，自乗算を正規基底（normal basis）がもつ巡回性を利用して $\mathbb{F}_p$ 上の加算回数を減らすことにより高速化を図った．次節では，正規基底がもつ巡回性に加えて，木構造を用いた処理を行うことで，さらなる高速化を目指す．

3.2.3 Type–$\langle 6, 5\rangle$ CVMA の改良

Type–$\langle 6, 5\rangle$ CVMA の入出力を式 (7) のように決めると，出力 Z は式 (8) のようになる．

$$\text{入力：} X = \sum_{i=0}^{m-1=4} x_i \gamma^{p^i}, \quad Y = \sum_{i=0}^{m-1=4} y_i \gamma^{p^i} \in \mathbb{F}_{p^5} \quad (x_i,\ y_i \in \mathbb{F}_p) \quad (7a)$$

$$\text{出力：} Z = X \cdot Y = \sum_{i=0}^{m-1=4} z_i \gamma^{p^i} \quad (z_i \in \mathbb{F}_p) \quad (7b)$$

表 3: TBF $\mathbb{F}_{p^5}$ および type–X AOPF $\mathbb{F}_{p^5}$ における演算のコスト

拡大体	TBF $\mathbb{F}_{p^5}$		type–$\langle 6,5\rangle$ GNB で構成された type–X AOPF $\mathbb{F}_{p^5}$		
乗算・自乗算アルゴリズム	Schoolbook 法 [14]	Karatsuba 法 [14]	従来の CVMA	SCIS2009 で最適化した CVMA [11]	本稿で最適化した CVMA
乗算	$(25,0,24,0,0)^{\dagger\ddagger}$	$(14,0,53,0,0)^{\dagger\ddagger}$	$(15,0,80,0,0)^{\ddagger}$	$(15,0,50,10,0)^{\ddagger}$	$(15,0,54,0,0)^{\ddagger}$
自乗算	$(9,5,14,10,0)^{\dagger\ddagger}$	$(14,0,29,5,0)^{\dagger\ddagger}$	$(15,0,70,0,0)^{\ddagger}$	$(15,0,40,10,0)^{\ddagger}$	$(15,0,44,0,0)^{\ddagger}$
Frobenius 写像	$(20,0,20,0,0)^{\ddagger}$		$(0,0,0,0,0)^{\ddagger}$		
逆元演算	$(120,0,112,0,1)^{\dagger\ddagger}$	$(98,0,170,0,1)^{\dagger\ddagger}$	$(44,0,188,0,1)^{\ddagger}$	$(44,0,124,24,1)^{\ddagger}$	$(44,0,136,0,1)^{\ddagger}$

† 法多項式として x^5-x-1 を用いた場合である．　‡ (a,b,c,d,e) は $aM_1+bS_1+cA_1+dD_1+eI_1$ を示す．

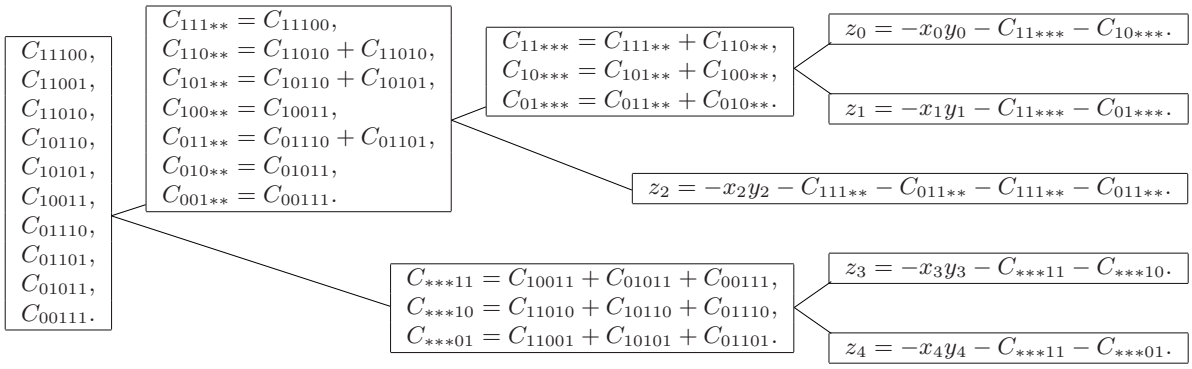

図 1: 木構造的な処理を取り入れた式 (9) の計算過程

$$\begin{aligned} z_0 = &- x_0y_0 - 2R_{01} - R_{03} - R_{12} - 2R_{13} \\ &- R_{14} - 2R_{23} - 2R_{24} - R_{34} \end{aligned} \tag{8a}$$

$$\begin{aligned} z_1 = &- x_1y_1 - R_{02} - 2R_{03} - R_{04} - 2R_{12} \\ &- R_{14} - R_{23} - 2R_{24} - 2R_{34} \end{aligned} \tag{8b}$$

$$\begin{aligned} z_2 = &- x_2y_2 - R_{01} - R_{02} - 2R_{03} - 2R_{04} \\ &- R_{13} - 2R_{14} - 2R_{23} - R_{34} \end{aligned} \tag{8c}$$

$$\begin{aligned} z_3 = &- x_3y_3 - 2R_{01} - 2R_{02} - R_{04} - R_{12} \\ &- R_{13} - 2R_{14} - R_{24} - 2R_{34} \end{aligned} \tag{8d}$$

$$\begin{aligned} z_4 = &- x_4y_4 - R_{01} - 2R_{02} - R_{03} - 2R_{04} \\ &- 2R_{12} - 2R_{13} - R_{23} - R_{24} \end{aligned} \tag{8e}$$

$$R_{ij} = (x_i - x_j)(y_i - y_j) \tag{8f}$$

ここで，式 (8) に対して $-x_iy_i$ を除いて考えると，R_{ij} の係数と z_l の関係は式 (9) のような行列式として表すことができる．

$$\left[\begin{array}{c|cccccccccc} & R_{01} & R_{02} & R_{03} & R_{04} & R_{12} & R_{13} & R_{14} & R_{23} & R_{24} & R_{34} \\ \hline z_0 & 2 & 0 & 1 & 0 & 1 & 2 & 1 & 2 & 2 & 1 \\ z_1 & 0 & 1 & 2 & 1 & 2 & 0 & 1 & 1 & 2 & 2 \\ z_2 & 1 & 1 & 2 & 2 & 0 & 1 & 2 & 2 & 0 & 1 \\ z_3 & 2 & 2 & 0 & 1 & 1 & 1 & 2 & 0 & 1 & 2 \\ z_4 & 1 & 2 & 1 & 2 & 2 & 2 & 0 & 1 & 1 & 0 \end{array}\right] \tag{9}$$

図 2 のように式 (9) の行列を分配すると，それぞれの行列の列ベクトルの値が一対一に対応する．このような対応関係は正規基底の巡回性によって生じたものである．ここで，その対応関係にしたがって，次式のように R_{ij} を結合したもの（コンポーネント）を考える．ただし，コンポーネントの添字は列ベクトルの値を意味する．

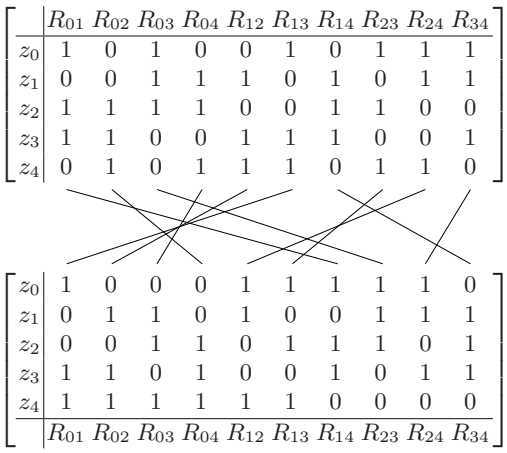

図 2: 式 (9) の行列の分配

$$C_{10110} = R_{01} + R_{14}, \quad C_{00111} = R_{02} + R_{04} \tag{10a}$$

$$C_{11100} = R_{03} + R_{13}, \quad C_{01101} = R_{04} + R_{03} \tag{10b}$$

$$C_{01011} = R_{12} + R_{02}, \quad C_{10011} = R_{13} + R_{01} \tag{10c}$$

$$C_{01110} = R_{14} + R_{34}, \quad C_{10101} = R_{23} + R_{13} \tag{10d}$$

$$C_{11001} = R_{24} + R_{12}, \quad C_{11010} = R_{34} + R_{24} \tag{10e}$$

これらのコンポーネントに対し，図 1 のような木構造を用いた処理行うことで，type–$\langle 6,5\rangle$ CVMA の計算コストは図 3 のようになる．

表 5: p =196–bit のときの TBF $\mathbb{F}_{p^5}$ および type–X AOPF $\mathbb{F}_{p^5}$ における演算の実装結果

拡大体	TBF $\mathbb{F}_{p^5}$		type–$\langle 6,5\rangle$ GNB で構成された type–X AOPF $\mathbb{F}_{p^5}$		
乗算・自乗算アルゴリズム	Schoolbook 法 [14]	Karatsuba 法 [14]	従来の CVMA	SCIS2009 で最適化した CVMA [11]	本稿で最適化した CVMA
乗算	**7.2** μsec	7.4 μsec	8.0 μsec	7.5 μsec	**7.3** μsec
自乗算	**5.7** μsec	6.5 μsec	7.6 μsec	7.0 μsec	**6.9** μsec
逆元演算	**45.8** μsec	45.9 μsec	30.0 μsec	28.9 μsec	**28.4** μsec

本節で，最適化したアルゴリズムは $p = 252$ の Freeman 曲線の場合でもそのまま適用できる．また，$p = 234$ の Freeman 曲線の場合は木構造を用いた処理のみ，適用することができる．

4 実装結果

今回最適化した CVMA の効果を表 5 に，p= 196–bit の Freeman 曲線を用いた Xt–Xate ペアリングおよび R–ate ペアリングの実装結果を表 6 に示す．ただし，今回の実装では [11] での実装と同様に剰余計算の削減を行った．また，実装環境は表 7 に示すとおりである．表 6 より，type–X AOPF $\mathbb{F}_{p^5}$ を用いた方が効率の良くペアリングを実装できるといえる．一方，p =149–bit，234–bit，252–bit の Freeman 曲線を用いた場合の実装については本稿では割愛し，CSS2009 の発表会場で結果を示す．

表 6: ペアリングの実装結果

$\mathbb{F}_{p^5}$	TBF†		type–X AOPF†	
ペアリング	Xt–Xate	Xt–R–ate	Xt–Xate	Xt–R–ate
Miller のアルゴリズム	7.32 ms	7.64 ms	6.64 ms	6.83 ms
最終べき	3.40 ms		3.45 ms	
計	11.3 ms	10.1 ms	10.1 ms	10.3 ms

†乗算・自乗算アルゴリズムには表 5 で最速だったものを使用した．

表 7: 実装環境

CPU	Pentium4 3.00GHz
二次キャッシュサイズ	512 KB
OS	Linux 2.6.27
プログラミング言語（コンパイラ）	C (gcc 4.3.2)
多倍長計算ライブラリ	GNU MP 4.2.4 [16]

謝辞

本研究は「文部科学省科学研究費補助金若手（B）20760241」の助成を受けて行われた．

参考文献

[1] R. Sakai, K. Ohgishi, and M. Kasahara, “Cryptosystems based on pairings,” SCIS2000, Okinawa, pp. 26–28, Jun. 2000.

[2] D. Boneh, X. Boyan, and H. Shacham, “Short group signatures,”, Proc. of Crypto2004, Lect. Notes Comput. Sci., Vol. 3152, pp. 41–55, 2004.

[3] 酒見由美，加藤英洋，野上保之，森川良孝，“整数変数 χ を用いて改良したクロスツイスト Ate ペアリング,” CSS2008, pp. 97–102, Oct. 2008.

[4] E. Lee, H. S. Lee, and C. M. Park, “Efficient and Generalized Pairing Computation on Abelian Varieties,” ePrint, No. 040, 2008.

[5] A. Miyaji, M. Nakabayashi, S. Takano, “New explicit conditions of elliptic curve traces for FR–reduction,” IEICE Trans. Fundam., Vol. E84, No. A(5), pp. 1234–1243, 2001.

[6] P. S. L. M. Barreto, and M. Naehrig, “Pairing-Friendly Elliptic Curves of Prime Order,” SAC 2005, LNCS, Vol. 3897, pp. 319–331, 2005.

[7] D. Freeman, “Constructing Pairing–Friendly Elliptic Curves with Embedding Degree 10,” ePrint, No. 026, 2006.

[8] D. Freeman, “Constructing Pairing-Friendly Elliptic Curves with Embedding Degree 10,” LNCS, Springer Berlin/Heidelberg, Vol. 4076, pp. 452–465, Oct. 2006.

[9] H. Cohen and G. Frey, “Handbook of elliptic and hyperelliptic curve cryptography,” Chapman & Hall/CRC, 2005.

[10] 加藤英洋，吉田知輝，野上保之，森川良孝，“Gauss Period Normal Basis を用いた拡大体上乗算に関する一考察”，CSS2007，pp. 477–482，Oct. 2007.

[11] 根角健太，柳枝里佳，吉田知輝，那須弘明，野上保之，森川良孝，“Freeman 曲線を用いた Xate ペアリングに適した拡大体の構成法”，SCIS2009, CD–ROM, Jan. 2009.

[12] A. J. Devegili, M. Scott, and R. Dahab, “Implementing Cryptographic Pairings over Barreto–Naehrig Curves,” LNCS, Springer Berlin/Heidelberg, Vol. 4575, pp. 197–207, 2007.

[13] T. Itoh and S. Tsujii, “A Fast Algorithm for Computing Multiplicative Inverses in GF(2^m) Using Normal Bases,” Inf. and Comp., Vol. 78, pp. 171-177, 1988.

[14] R. Dahab, A. J. Devegili, C. Ó hÉigeartaigh, and M. Scott, “Multiplication and Squaring on Pairing–Friendly Fields,” ePrint, No. 471, 2006.

[15] S. Gao, “Abelian Groups, Gauss Periods and Normal Bases,” Finite Fields Appl. 7, No.1, pp.148–164, 2001.

[16] GNU Multiple Precision Arithmetic Library, available at “http://gmplib.org”.